

Display Games

Mission 4

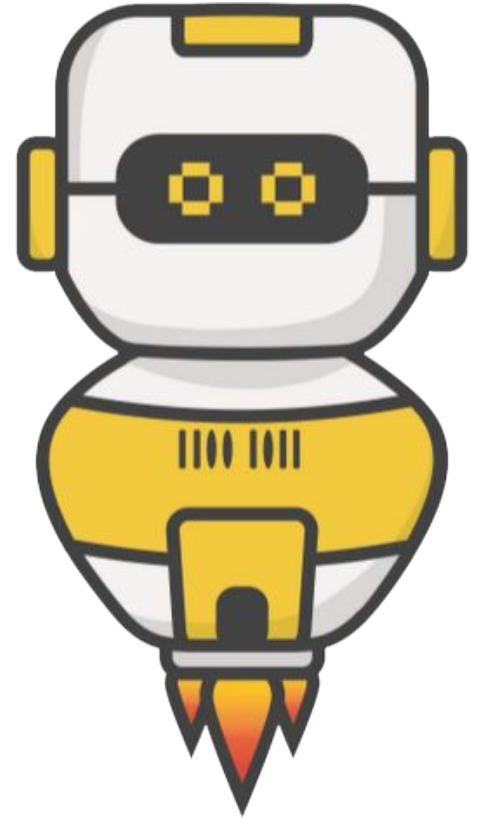


FIRIA LABS

Pre-Mission Preparation

In the Mission 4 log, answer the pre-mission preparation questions:

- Other than a computer or cellphone, what are some things that have a screen?
- What are things you might want to display on a screen?

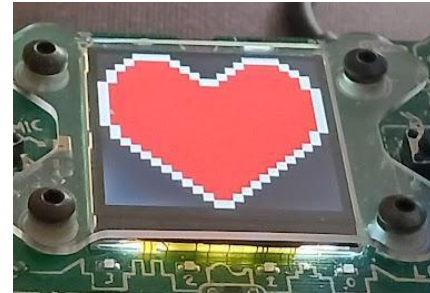


Objective #1: Back to the display

In Mission 2, your program displayed an image. The first image you displayed was a HEART.

- You will practice displaying an image on the LCD screen

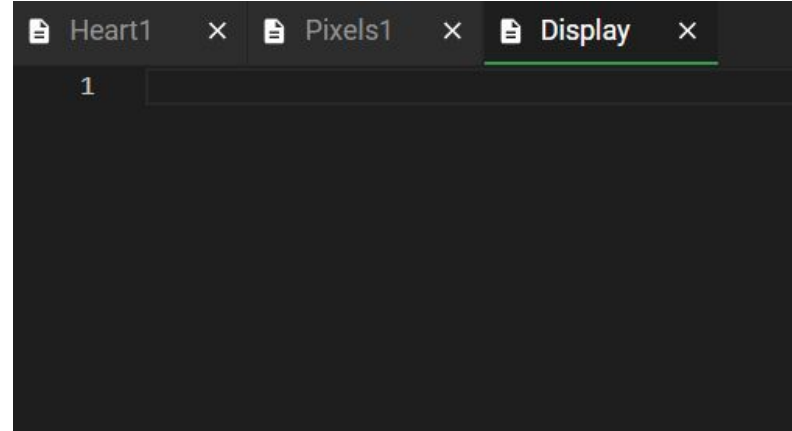
```
Heart1 x Pixels1 x
1 from codex import *
2 display.show(pics.HEART)
```



Mission Activity #1

DO THIS:

- Create a new file named **Display**
- Click the **File** menu button
- Select “New File...”
- Name the file **Display**
 - *no spaces in a file name*
- Click **Create**



Mission Activity #1

DO THIS:

- Add two lines of code to display a PLANE
- Run your code

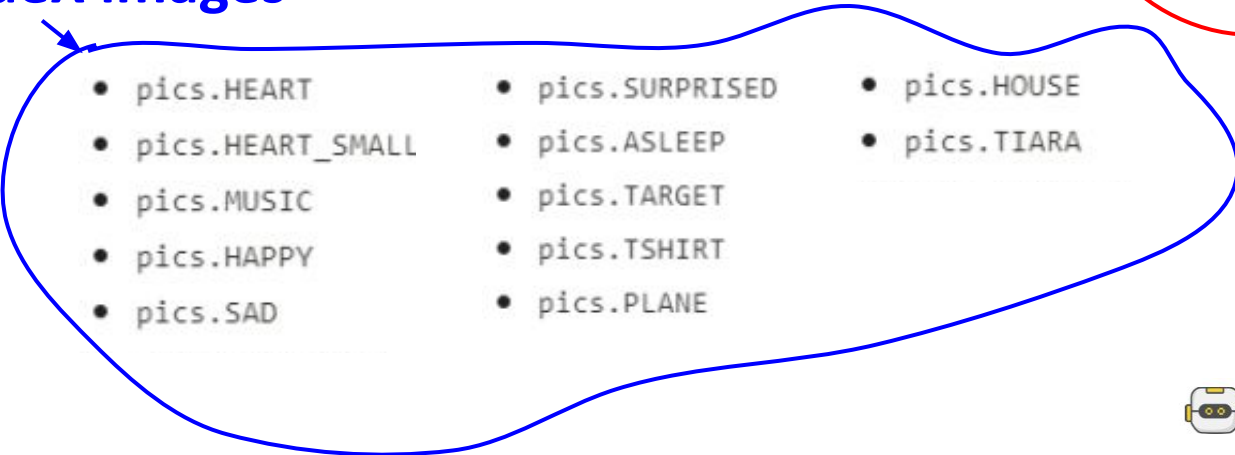
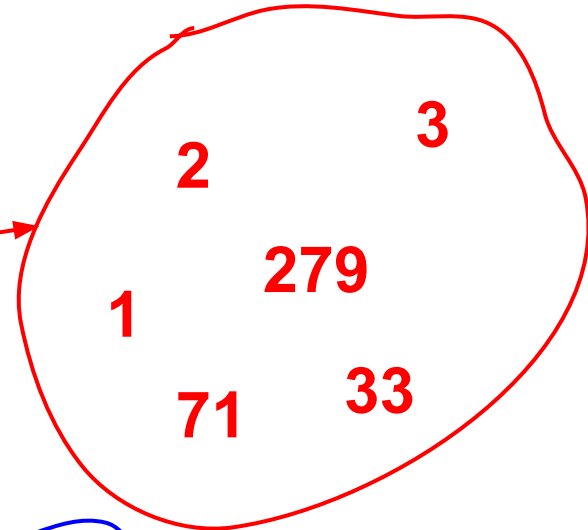
```
Heart1 × Pixels1 × Display ×
1 from codex import *
2 display.show(pics.PLANE)
3
4
```



Objective #2: Text messages

Computers work with different **types** of data.
So far you have worked with:

- **Integers** (counting numbers)
- **CodeX images**



Mission Activity #2

DO THIS:

- Go to the Mission Log
- Write examples of different data types
- Change the code in the `display.show()` function to display the text “Ahoy”

```
Display x
1 from codex import *
2 display.show("Ahoy")
3
```



Objective #3: Good with numbers?

A computer is very good at doing math.

- When you define a **variable**, you assign it a value
- So far you assigned a **literal** value
- You can also assign a value by doing math

```
num = 2 + 2
```



Mission Activity #3

Use a simple calculation to assign a value to a variable

DO THIS:

- Add a line of code that uses the assignment statement shown on the last slide

```
num = 2 + 2
```

- Use the `display.show()` statement to show the `num` variable

```
Display ×
1 from codex import *
2 num = 2 + 2
3 display.show(num)
4
```

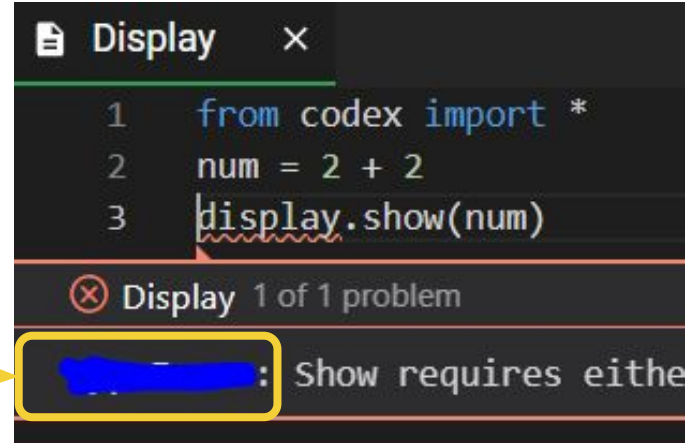


Mission Activity #3

The code caused an error

DO THIS:

- Go to your Mission Log and write the error message



```
Display x
1 from codex import *
2 num = 2 + 2
3 display.show(num)
Display 1 of 1 problem
[redacted]: Show requires either
```



Objective #4: Converting types

Why does `display.show(num)` not work?

- `display.show("Ahoy")` works
 - `"Ahoy"` is a type **string**
- `display.show(pics.HEART)` works
 - `pics.HEART` is a type **CodeX image**
- `display.show(num)` does not work
 - `num` is a type **integer**



Objective #4: Converting types

Why does `display.show(num)` not work?

It doesn't work with an **integer**, but it will work with a **string** -- the **type** is wrong.

- If an integer is converted, or changed, to a string, then `display.show()` will work -- no error
- Python has a function that will convert (change) any value to a **string**
 - `str()`

EXAMPLES:

For each of these examples, the value in the (parenthesis) is changed to a

“string”

- `str(1)`
- `str(num)`



Mission Activity #4

Modify your code by using the **str()** conversion function

DO THIS:

- Change the `display.show(num)` code to use the **str()** function
 - Be careful to match your parenthesis
- Run your code

```
Display ×  
1 from codex import *  
2 num = 2 + 2  
3 display.show(str(num))  
4
```



Objective #5: Second show message

Can you display two messages?

In Mission 3, you tried showing two (or four) different colors in a pixel. This didn't work until you slowed down the program by using a `sleep()`.

What do you think will happen if you try to display two messages?



Mission Activity #5

DO THIS:

- What do you think will happen if you try to print two messages?
- Go to the Mission Log and write your prediction
- Then change your code to display two messages:

```
Display ×  
1 from codex import *  
2 display.show("Hello")  
3 display.show("World")  
4
```



Objective #6: Printing text

The `display.show()` command will only show one thing at a time. So, just like the pixels, the second thing is displayed on top of the first thing.

- CodeX has another way to display a **string**
- Use **`display.print("string")`**
- All **`display.print("string")`** messages will be displayed, one after another -- each on its own line



Mission Activity #6

Change your code to **print** the **strings**.

DO THIS:

- Change the `display.show()` command to `display.print()`

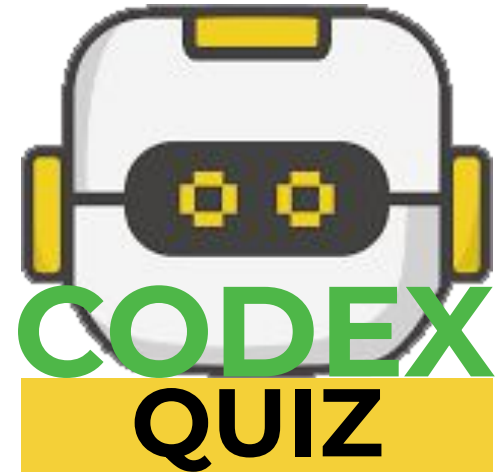
```
Display x
1 from codex import *
2 display.print("Hello")
3 display.print("World")
4
```



Typed and Printed quiz

So far you have learned about Python data types and the `display.print()` command.

- Answer the quiz questions



Objective #7: Branching

During the next objectives, you will create a button-pressing game. Here are the parts of the game:

1. Display a button to press.
2. Press and hold the button. You will have one second.
3. If the correct button is pressed, light a pixel GREEN, otherwise light the pixel RED



Objective #7: Branching

Step #3 is a new concept -- branching.

- Branching is when the computer makes a choice between two things.
- Here is an example of branching.
- Notice the indenting -- this is very important!

3. If the correct button is pressed, light a pixel GREEN, otherwise light the pixel RED

```
if pressed:  
    pixels.set(0, GREEN)  
else:  
    pixels.set(0, RED)
```



Objective #7: Branching

Take a closer look at branching:

This is a condition that will be True or False

```
if pressed:  
    pixels.set(0, GREEN)  
else:  
    pixels.set(0, RED)
```

The IF part will run when the condition is True

Notice the colon(:) after the **if** and **else**

The ELSE part will run when the condition is False



Objective #7: Branching

```
if pressed:
    pixels.set(0, GREEN)
else:
    pixels.set(0, RED)
```

In this example, `pressed` will be either **True** or **False** (no “quotations”)

This is a data **type**: **Boolean**

Now you know four data types:

- **Integer**
 - 1, 54, 720
- **CodeX image**
 - `pics.HEART`, `pics.MUSIC`
- **String**
 - “Hello”, “Press A”, “cake”
- **Boolean**
 - True, False



Mission Activity #7

The best way to learn about branching is to try it:

DO THIS:

- Delete most of your code and type the code to the right
- Run the code
 - Do you see a GREEN light?
- Change line 5 to **pressed = False**
- Run the code
 - Do you see a RED light?

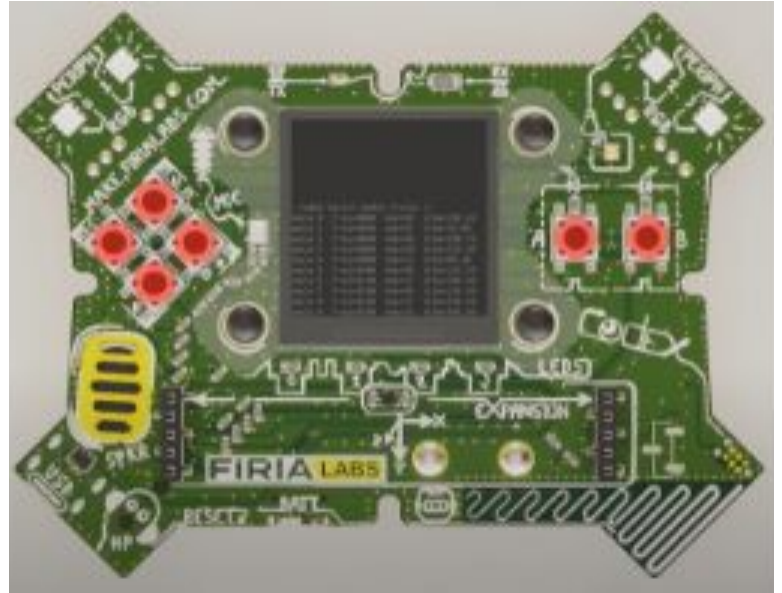
```
Display ×
1  from codex import *
2  from time import sleep
3
4  sleep(1)
5  pressed = True
6  if pressed:
7      pixels.set(0, GREEN)
8  else:
9      pixels.set(0, RED)
10
```



Objective #8: Button hunting

The game will use any four of the six buttons.

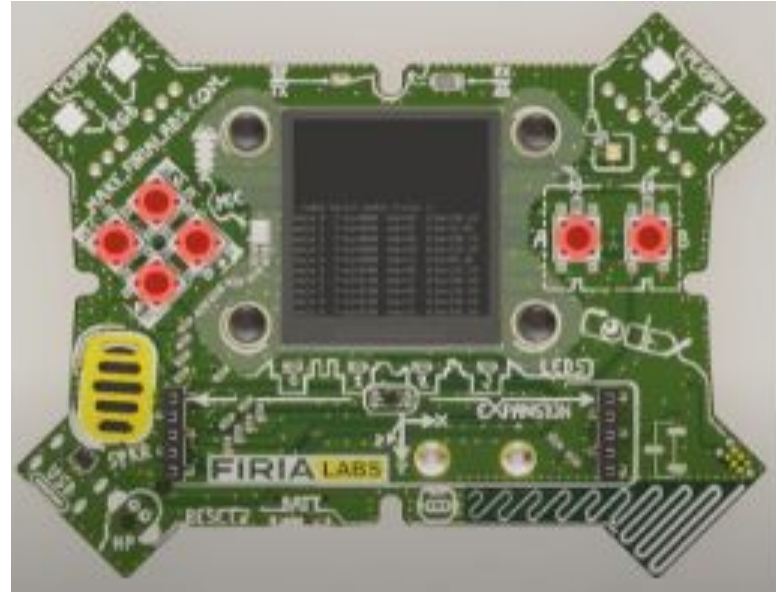
- Look at the picture of the CodeX.
- Can you find all 6 buttons?



Mission Activity #8

DO THIS:

- Close the instructions panel
- Use the camera to rotate the CodeX until you see the front
- Click on all 6 buttons



Objective #9: Gamer input

There are 2 ways to check for a button press:

- `buttons.was_pressed(BTN_A)`
 - Checks to see if button A was pressed since the last check
- `buttons.is_pressed(BTN_A)`
 - Checks to see if button A is currently pressed

Any of the 6 buttons can be checked in ().



Mission Activity #9

```
Display x
1 from codex import *
2 from time import sleep
3
4 display.show("Press Button A")
5 sleep(1)
6 pressed = buttons.is_pressed(BTN_A)
7 if pressed:
8     pixels.set(0, GREEN)
9 else:
10    pixels.set(0, RED)
11
```

For this game, you will check for currently pressed

DO THIS:

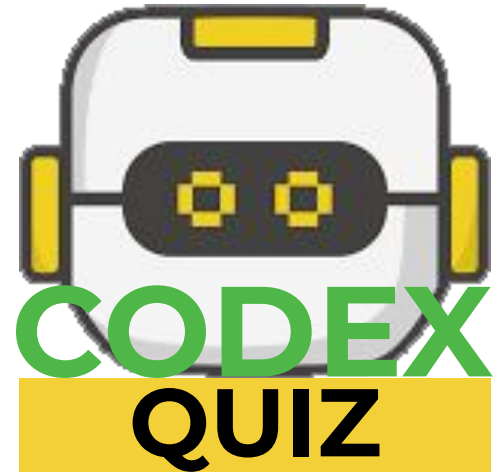
- Add line 4
- Change line 6
- Run the code and press Button A
- Run the code again and do not press Button A
- Do you get the results you expect?



Buttons and Branching quiz

You have been learning about branching and checking for pressed buttons.

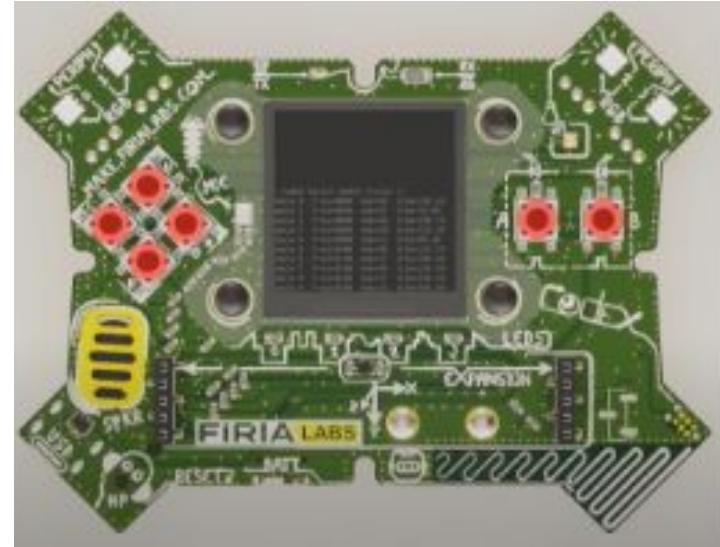
- Answer the two quiz questions



Objective #10: For the win!

Now just check a few more buttons and you have a serious twitch game!

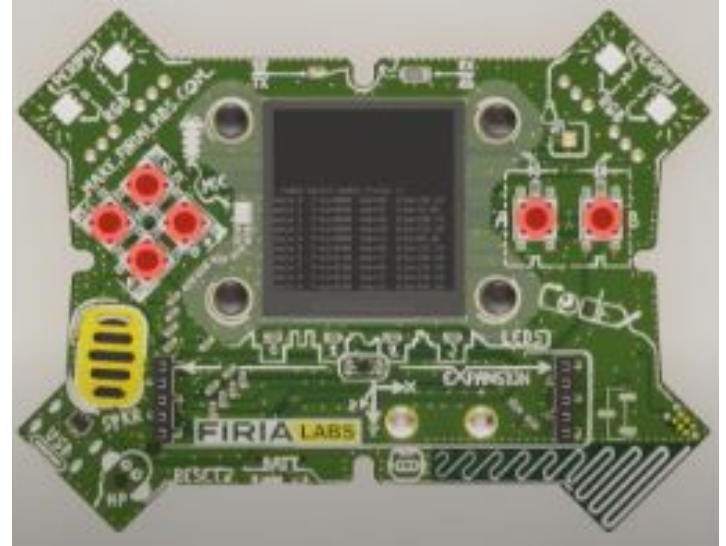
- You can use whatever buttons you want
- You have 6 buttons to choose from:
 - BTN_A, BTN_B,
BTN_L, BTN_R, BTN_U, BTN_D
- What 4 buttons do you want to use for your game?



Mission Activity #10

DO THIS:

- Go to the Mission Log and record the four buttons you will use for your game



Mission Activity #10

DO THIS:

- In your code, copy line 4 through line 10
- Paste the code below line 10
- Change Button A to the second button you want to use
 - Change it in the `display.show()`
 - Change it in `buttons.is_pressed`
- Change the pixel from 0 to 1

```
Display x
1 from codex import *
2 from time import sleep
3
4 display.show("Press Button A")
5 sleep(1)
6 pressed = buttons.is_pressed(BTN_A)
7 if pressed:
8     pixels.set(0, GREEN)
9 else:
10    pixels.set(0, RED)
11
```



Mission Activity #10

DO THIS:

- Paste the code again, below the current code
- Change the button to the third button you want to use
 - Change it in the `display.show()`
 - Change it in `buttons.is_pressed`
- Change the pixel to 2

```
1  from codex import *
2  from time import sleep
3
4  display.show("Press Button A")
5  sleep(1)
6  pressed = buttons.is_pressed(BTN_A)
7  if pressed:
8      pixels.set(0, GREEN)
9  else:
10     pixels.set(0, RED)
11
12  display.show("Press Button L")
13  sleep(1)
14  pressed = buttons.is_pressed(BTN_L)
15  if pressed:
16     pixels.set(1, GREEN)
17  else:
18     pixels.set(1, RED)
19
20
```



Mission Activity #10

DO THIS:

- Paste the code one more time, below the current code
- Change the button to the fourth button you want to use
 - Change it in the display.show()
 - Change it in buttons.is_pressed
- Change the pixel to 3

```
1 from codex import *
2 from time import sleep
3
4 display.show("Press Button A")
5 sleep(1)
6 pressed = buttons.is_pressed(BTN_A)
7 if pressed:
8     pixels.set(0, GREEN)
9 else:
10    pixels.set(0, RED)
11
12 display.show("Press Button L")
13 sleep(1)
14 pressed = buttons.is_pressed(BTN_L)
15 if pressed:
16    pixels.set(1, GREEN)
17 else:
18    pixels.set(1, RED)
19
20 display.show("Press Button B")
21 sleep(1)
22 pressed = buttons.is_pressed(BTN_B)
23 if pressed:
24    pixels.set(2, GREEN)
25 else:
26    pixels.set(2, RED)
27
```



Mission Activity #10

At this point you should have code for the four buttons you chose.

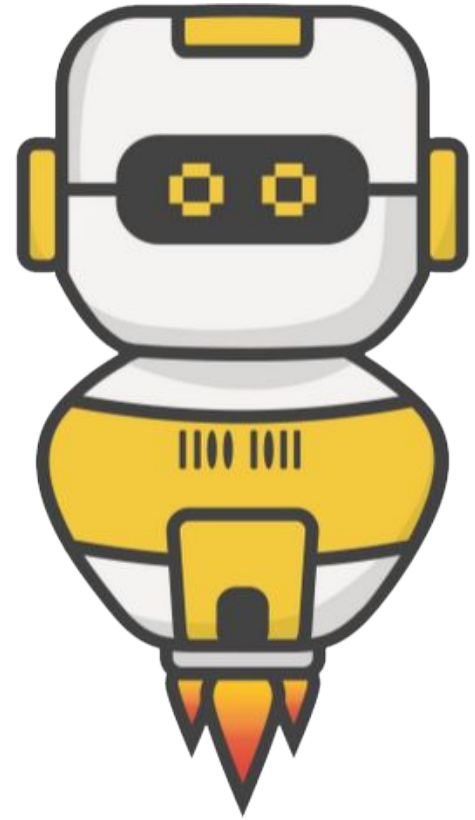
DO THIS:

- Run the code
- If you have any errors, fix them
- Try pressing all the buttons and getting all green lights
- Try the code again, missing some of the buttons
- Do you get the results you expect?
- Make any changes you need to so that your program works correctly
- Have someone else try your game



Post-Mission Reflection

- Read the “completed mission” message and click to complete the mission
- Complete the Mission 4 Log



Clearing your CodeX

Go to FILE -- BROWSE FILES
Select the “**Clear**” file and open it
Run the program to clear the CodeX



FIRIA LABS